DAVID BRUBAKER,
CONTRIBUTING EDITOR

# What exactly is "neuro-fuzzy"?

My father is fond of telling the story (I think he read it in *Reader's Digest*) about a sailor on a Navy ship who wanted to yank the chain of the chief petty officer (CPO) in charge of one of the ship's tool lockers. The CPO had just finished a two-hour lecture on what tools were available, their correct names, and what each was used for. The sailor requested a wrench from the locker. Indignant, the CPO demanded, "Seaman, didn't you hear a word I said? What kind of wrench do you want: an open-end wrench, a box-end wrench, a combination wrench, a crescent wrench, a pipe wrench, a socket wrench," and so on, listing the available types of wrenches. After the barrage, the sailor replied, "What type really doesn't matter, Chief, as long as it's big. I'm going to use it as a hammer."

**Often, a single tool is inadequate, and you must use two or more tools together to solve a problem.**

The right tool for the job. As engineers, we must also be aware of the tools that are available and how to use them. I hope this column has encouraged some of you to think of various fuzzy methods as engineering tools, appropriate for what appears to be a wide range of jobs.

Often, however, a single tool is inadequate, and you must use two or more tools together to solve a problem. In this column and the next two, I will discuss combining two engineering tools available to complex-system designers: fuzzy rule bases and neural networks. The media has dubbed this hybrid "neuro-fuzzy."

First let's get a feel for different kinds of neuro-fuzzy combinations. You can divide the marriage of neural nets and fuzzy rule bases into three categories: systems with separate, interactive components; systems in which one technology assists the design of the other; and systems in which the source technologies "fuse" to form a new technology.

An interactive-component system consists of nodes that execute independently from each other and generate data that is either output from the system or sent as inputs to other nodes. Both an object-oriented system and a multitask, data-driven system are examples of an interactive component architecture. When some of the components are fuzzy rule bases and others are neural networks, you have a neuro-fuzzy hybrid.

Using a neural-network assistant to design a fuzzy rule base is the most common neuro-fuzzy hybrid to date. You use this approach when plenty of data exist for system definition, and the approach is most effective when the resulting rule base is complex enough to hinder an unassisted design. As a design assistant, a neural net can translate training data into input- and output-membership functions for a given set of rules, or you can use a more complex neural net to define rules.

The third hybrid structure, a fused-component system, exists when you meld two or more components to form a single (you hope) synergistic structure.

For example, I consider the fuzzy cognitive map (**References 1** and **2**) to be a fused neuro-fuzzy system,

Choosing when to use a fuzzy rule base and when to use a neural network—whether for a given component in a neuro-fuzzy hybrid architecture or as the principal component in a nonhybrid system—depends on the relative strengths of the two competing technologies. Before discussing these strengths, we need a little background.

Both fuzzy rule bases and neural nets are function approximators. This term means that both map their inputs to a specific set of outputs and that both are theoretically able to approximate any arbitrarily complex function. In their standard configurations, both are also deterministic (the same set of inputs always generates the same output value) and time-invariant (the generated response function is independent of time).

When taken as black boxes—that is,

PHOTOGRAPHY / LIANE ENKELIS

ignoring the internal structural details—the primary differences between a fuzzy rule base and a neural network are as follows:

1. A domain expert "programs" the operation of a fuzzy rule base by listing rules and defining the meanings of the rule terms with fuzzy sets (which are represented as membership functions). In contrast, the neural net's training occurs when internal interconnection weights are altered, causing the neural net to match the operation represented by a set of input/output data points. Viewed in terms of how humans learn, a rule base is "taught by a teacher" (the domain expert), and a neural net "learns from experience" (implicit in the training data). The quality of operation of a neural network is, therefore, dependent on the quality of data, and the quality of operation of the fuzzy-rule-based system is dependent on the quality of the expert's knowledge and her or his ability to express it. These dependencies are true both for initial designs and for modifications of existing designs.

2. A neural network can more easily handle increased numbers than a fuzzy-rule-based system. A fuzzy rule base requires one dimension in its rule matrix for each input. Therefore, as the number of inputs linearly increases, the size of the rule base exponentially increases. The required increase in the size of a neural network in response to an increase in the number of inputs depends not only on how the network is structured. Rather, its structure depends on the complexity of the response function, the capacity of the current neural-net configuration, and the tolerance of the required outputs. Adding a single input to a neural network may require as little as adding a single node to its input layer or as much as adding one or more new inner layers.

3. In general, for a small system (in which the number of inputs is fewer than or equal to three and the shape of the response function is only moderately complex), runtime execution of a fuzzy-rule-based system is faster than that of a neural network. As the number of inputs increases—because the size of the neural net typically must increase less than the size of the fuzzy ation of a neural net is ultimately less than the time to execute one iteration of a fuzzy rule base. The crossover point depends on the processor you use, the language you use, the amount of available memory, the quality of the code, and the tricks you use for either structure.

4. A fuzzy rule base system's design is self-documenting. Membership functions define the linguistic variables (words) contained in rules, and rules describe the operation of the system using words. Even in a complex system, rules and membership functions explain the "reasoning" behind the operation of a fuzzy rule base over its input space. Conversely, the weights of many potential internal connections hold the reasoning behind the operation of a neural net. Even with a single hidden layer, the meanings of these weights are difficult or impossible to relate to actual operation.

This column is all rather abstract, but overviews tend to be that way. Next time, I shall discuss an actual example: a system with three fuzzy rule bases and a single neural network. In addition to presenting structural details, I shall also list the sequence of decisions that led to the use of this hybrid structure. In a subsequent column, I will present a simple example of how you can use a neural network to design a fuzzy rule base. EDN

## References

1. Brubaker, David, "Fuzzy cognitive maps," *EDN*, April 11, 1996, pg 209.
2. Brubaker, David, "More on fuzzy cognitive maps," *EDN*, April 25, 1996, pg 213.

**VOTE**

Please use the Information Retrieval Service card to rate this article (circle one):

| High Interest | Medium Interest | Low Interest |
|---|---|---|
| 586 | 587 | 588 |

*David Brubaker is a consultant in fuzzy-system design. You can reach him at Huntington Advanced Technology, 883 Santa Cruz Ave, Suite 31, Menlo Park, CA 94025-4608 or on the Internet at: brubaker@cup.portal.com.*